REMARKS

I. Introduction

In response to the Office Action dated August 9, 2007, claims 5 and 13 have been amended. Claims 1-18 remain in the application. Re-examination and re-consideration of the application, as amended, is requested.

II. Claim Amendments

Applicants' attorney has made amendments to the claims as indicated above. These amendments were made solely for the purpose of clarifying the language of the claims, and were not required for patentability or to distinguish the claims over the prior art.

III. Specification

In paragraph (3) of the Office Action, the Patent Office noted that trademarks should be capitalized wherever it appears and be accompanied by generic terminology. Similar to the prior response, Applicants again note that the trademarks Netscape Navigator™ and Microsoft Internet Explorer™ only appear once each in the specification in paragraph [0023] on page 7 of the originally filed specification. In that paragraph, the terms are already in full capitalization and easily distinguishable from the remaining text. Accordingly, Applicants submit that efforts have been made to prevent their use in any manner which might adversely affect their validity as trademarks. In view of the above, Applicants respectfully request withdrawal of the objections.

IV. Office Action Subject Matter Rejection

In paragraphs (4)-(5), the Office Action rejects claims 5-8 and 13-16 under 35 U.S.C. § 101 as being directed to non-statutory subject matter. The Applicants have amended claims 5 and 13.

Applicants believe that the claims currently describe statutory subject matter. Should issues still remain in this regard, the Applicants request that the Examiner indicate how the rejection can be overcome and how problems may be resolved, in accordance with the directives of the Examination Guidelines for Computer-Related Inventions. See Guidelines II M.P.E.P. § 2106. Specifically, should it be necessary, the Applicants request that the Examiner identify features of the invention

that would render the claimed subject matter statutory if recited in the claim. See Guidelines IV, M.P.E.P. § 2106.

V. Prior Art Rejections

In paragraphs (6)-(7) of the Office Action, claims 1-17 were rejected under 35 U.S.C. §103(a) as being unpatentable over Hirsch, U.S. Patent No. 6,915,301 (Hirsch).

Specifically, claims 1, 5, 9, 13, and 17-18 were rejected as follows:

> Claim 1: Hirsch discloses a method and computer system for dynamic properties of software objects comprising:
>
> a. receiving a reference to an object instance having a dynamic property that is created at runtime for the object instance on a per-instance basis and is not stored with the object (column 11, lines 37-58);
>
> Hirsch discloses retrieving reference to a property source instance (data source) from an association between the object and the property source instance (binding), wherein the property source instance creates and supplies to dynamic property (column 4, lines 11-34/column 11, lines 37-59), but does not explicitly disclose creating and supplying an initial value for the dynamic property for/to the object instance. However, Hirsch does disclose data sources in each scene. Furthermore, providing initial values was a well-known technique in the art at the time the invention was made. Therefore it would have been obvious to one having ordinary skill in the art at the time the invention was made that the data sources may supply an initial value for the dynamic property for/to the object instances in Hirsch. One would have been motivated to create and supply an initial value for the dynamic property in order to initialize the property to an initial value at runtime.
>
> c. providing the reference to the object and the reference to the property source instance to a control which is configured to: (i) retrieve the dynamic property from the property source and (ii) display the property (object inspector window) in a user interface (column 11, 37-67).
>
> Claim 5: Hirsch discloses a method and computer system for dynamic properties of software objects comprising:
>
> a. an object instance having a dynamic property that is created at runtime for the object instance on a per-instance basis and is not stored with the object (column 11, lines 37-59);
>
> Hirsch discloses a property source instance wherein the property source instance creates and supplies the dynamic property (column 11, lines 37-59) but does not explicitly disclose creating and supplying an initial value for the dynamic property for/to the object instance. However, Hirsch does disclose data sources generate values in a scene, and objects and their properties are bound to data sources in each scene (column 4, lines 11-34/column 11, lines 37-59). Furthermore, providing initial values was a well-known technique in the art at the time the invention was made. Therefore it would have been obvious to one having ordinary skill in the art at the time the invention was made that the data sources may supply an initial value for the dynamic property for/to the object instances in Hirsch. One would have been motivated to create and supply an initial value for the dynamic property in order to initialize the property to an initial value at runtime.
>
> c. an association between the object and the property source instance (column 3, lines 8-27);
>
> d. a host configured to: (i) retrieve a reference to the object instance; (ii) retrieve a reference to the property source; (iii) provide the reference to the object and the reference to the property source to a control which is configured to (1) retrieve the dynamic property from the property source and (2) display the property in a user interface (column 11, lines 37-67).

Claim 9: Hirsch discloses a method and computer system for dynamic properties of software objects comprising receiving a first object having a first property wherein the first object provides a control (calendar control) that defines a first user interface for displaying and editing the first property, but does not explicitly disclose the control is an ActiveX control. However, Hirsch does disclose supporting ActiveX controls (column 12, lines 8-31). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to provide an ActiveX control to display and edit the first property. One would have been motivated to provide an ActiveX control to display and edit the first property in order to make the design more efficient.

b. creating a list of one or more object properties to be displayed, wherein the list includes the first property (column 12, lines 8-31);

c. instantiating the custom ActiveX control (column 12, lines 8-31);

d. displaying the object properties in the list, wherein the display of the first property comprises the first user interface defined by the instantiated ActiveX control, wherein the property may be edited through the first user interface (column 1, line 8-31).

Claim 13: Hirsch discloses a method and computer system for dynamic properties of software objects comprising:

a. one or more objects, wherein each object has one or more object properties (column 11, lines 37-59);

b. a property inspector configured to (i) interrogate the one or more objects to discover one or more object properties to be displayed; (ii) create a list of the one or more object properties to be displayed; (iii) instantiate and host one or more property editors (column 11, lines 37-59/column 12, lines 1-21);

Hirsch discloses one or more property editors wherein: (i) one of the property editors comprises a custom control specified by one of the objects, but does not explicitly disclose the control is an ActiveX control. However, Hirsch does disclose supporting ActiveX controls (column 12, lines 8-31). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to provide an ActiveX control specified by one of the objects. One would have been motivated to provide an ActiveX control specified by one of the objects in order to make the design more efficient.

Hirsch discloses (ii) the custom ActiveX control defines a custom graphical user interface for displaying and editing one of the object properties (column 12, lines 8-31).

Claim 17: Hirsch discloses a method and computer system for dynamic properties of software objects comprising:

a. an object instance of a class, wherein (i) an initial value for one or more static properties of the class are assigned at run time; and (ii) the object instance has a dynamic property that is generated by a property source instance at runtime for the object instance on a per-instance basis and are not stored with the object (column 12, lines 20-58/column 11, lines 37-59). Hirsch does not explicitly disclose an initial value is generated and supplied by a property source instance. However, Hirsch does disclose data sources generate values in a scene, and objects and their properties are bound to data sources in each scene. Furthermore, providing initial values was a well-known technique in the art a the time the invention was made. Therefore it would have been obvious to one having ordinary skill in the art at the time the invention was made that the data sources may supply an initial value for the dynamic property for/to the object instances in Hirsch. One would have been motivated to create and supply an initial value for the dynamic property in order to initialize the property to an initial value at runtime.

b. an association between either: (i) the object instance and the property source instance; (ii) the class and the property source instance (column 11, lines 37-61)

c. a user interface component that displays a collection of properties of the object instance including the one or more static properties and the dynamic property on a display device(column 11, lines 37-61), wherein the user interface component is configured to:

(i) retrieve a reference to the object instance (column 11, lines 49-63);
(ii) retrieve the one or more static properties form the object instance (column 11, lines 49-63);
(iii) access the association to determine the property source instance associated with the object instance (column 11, lines 37-61);
(iv) call a method of the determined property source instance with the reference to the associated object instance (column 11, lines 37-61);
(v) receive the dynamic property, from the property source instance, wherein the property source instance dynamically generated the dynamic property (column 11, lines 49-63). Hirsch does not explicitly disclose an initial value is generated and supplied by a property source instance. However, Hirsch does disclose data sources generate values in a scene, and objects and their properties are bound to data sources in each scene. Furthermore, providing initial values was a well-known technique in the art at the time the invention was made. Therefore it would have been obvious to one having ordinary skill in the art at the time the invention was made that the data sources may supply an initial value for the dynamic property for/to the object instances in Hirsch. One would have been motivated to create and supply an initial value for the dynamic property in order to initialize the property to an initial value at runtime.
(vi) display the static property on the display device (column 11, lines 49-63).

Claim 18: Hirsch discloses a method and computer system for dynamic properties of software objects as in Claim 9 above, further comprising custom controls are provided on a per-instance basins, but does not explicitly disclose the controls are ActiveX controls. However, Hirsch does disclose supporting ActiveX controls (column 12, lines 8-31). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to provide an ActiveX control on a per-property basis. One would have been motivated to provide an ActiveX control on a per-property basis in order to make the design more efficient.
c. two or more object properties are in the properties list (column 12, lines 8-31);
d. the two or more object properties are displayed in a list, wherein each of the two or more object properties are displayed using user interfaces defined by the custom ActiveX controls (column 12, lines 8-31).

Applicants traverse the above rejections. There are essentially two sets of claims. The first set includes independent claims 1, 5, and 17. The second set includes claims 9 and 13. Each set will be addressed separately herein.

Claims 1, 5, and 17

These independent claims provide the ability to create and display a dynamic property on a per object basis. As explicitly set forth in the claims, the dynamic property is created at runtime for an object instance on a per-instance basis and is not stored with the object. Referring to claim 1, the first limitation provides for retrieving a reference to an object instance. This element provides an instance of an object/class for which the dynamic property will be created.

The second limitation retrieves a reference to a property source instance. Applicants note that the limitation provides that it is an instance of the property source (i.e., it is an instance of an

object). Further the reference to this property source instance is obtained. The second limitation also provides that the reference is retrieved from "an association between the object and the property source instance". Thus, rather than retrieving a reference from a random location, it is retrieved from an entity identified as an "association" in the claims. Such an association can take a variety of forms (e.g., a mapping, a linked list, etc.). In addition, the second element provides that the property source instance creates and supplies the dynamic property and an initial value for the dynamic property for/to the object instance. Accordingly, when examining this claim limitation in combination with the first limitation, the claims explicitly provide that a property source instance creates (at run time) and supplies (at run time) a new property and an initial value for the new property to/for the object instance. Thus, rather than the object instance creating a value for one of its own properties at runtime (e.g., a name for the object instance), this claim element explicitly provides that a property itself AND a value for that property is created for one object (i.e., the object instance) by another object (i.e., the property source instance).

The third limitation provides the two references (i.e., the reference to the object instance and the reference to the property source instance) to a control. The third limitation further provides that the control is configured to retrieve the dynamic property from the property source instance and display the dynamic property in a user interface.

As can be seen by the above limitations, there are specific and detailed claim limitations that provide a precise structure and capability. More specifically, a property source instance creates a property (and a value for that property) for an object instance. Such a creation is performed dynamically during runtime.

The Office Actions rejects all of the claim limitations based on Hirsch. More specifically, Hirsch, Col. 11, lines 37-67 is used to reject the claims. Col. 11, lines 37-67 provides:

> Object properties consist of named attributes that define an object's appearance in terms of a functional expression. Access to these properties are provided through the Object Inspector. Their values are set at runtime based on the calculated result of the expressions. An expression may include the names of one or more data source columns which are automatically bound to a result row at runtime.
>
> Object property expressions result in one of the following base data types: Boolean, Numeric, String, Point, PointList, and Image. Derived Numeric types include Color, DateTime, Enum, Integer, and Percentage. Derived String types include FilePath (or URL) and FontName.
>
> Referring now to FIG. 2, the object inspector 30 is shown in more detail. The object inspector 30 provides two columns, an attribute name column 100 and a property column 110. The object being inspected in FIG. 2 has properties 112, 114, 116, 118, 120, 122, 124 and 126. Particularly, properties 114, 116, 118, 120, 124 and 126 are static constants. However, properties 112 and 122 are

> dynamic and are evaluated at run time. Due to the dynamic properties 112 and 122, to create a data driven application, the user only needs to enter the dynamic properties and code may be automatically generated. Thus, the data binding of the dynamic properties 112 and 122 is seamlessly integrated into the property sheet 30.
>
> The Object Inspector window 30 is a dockable control bar which displays an object's properties and events. The Object Inspector may be resized horizontally when docked and both vertically and horizontally when floating. The window may dock to the left or right of the world workspace when its location overlaps the docking site while being dragged.

As can be seen from this text, Hirsch completely and entirely fails to teach, describe, or suggest, explicitly or implicitly the creation of a property source instance for a property of a separate object instance. Hirsch further fails to teach, describe, or suggest the creation of a value for such a dynamic object. Applicants note that the term "dynamic" as set forth in the present claims refers to a property that is actually created and a value for that property is actually created dynamically. Hirsch does not utilize such a definition. Instead, Hirsch describes an object inspector that merely lists all of the properties of an object. Hirsch's "dynamic" property merely refers to properties 112 and 122 that are dynamic in the sense that they are evaluated at run time and are not static constants (such as properties 114, 116, 118, 120, 124, and 126). Thus, Hirsch's dynamic properties already exist for an object but are merely evaluated at run time. Such a teaching is not even remotely similar to the present claims which explicitly provide that a property source instance actually creates and supplies a dynamic property and a value for the dynamic property at run time while both the dynamic property and value are for a separate object instance.

The Office Action bypasses such a creation of the property and value elements and asserts that Hirsch does not explicitly disclose creating and supplying an initial value for a dynamic property to an object instance. Instead, the Action states that Hirsch discloses data sources that generate values in a scene, and objects and their properties are bound to data sources in each scene. However, once again, the claims are not directed towards merely generating values in a scene or binding objects and properties to data sources in a scene. Instead, the claims explicitly provide for a property source instance creating and supplying both a dynamic property and an initial value for the dynamic property to a separate object instance. In addition, a control is sued to retrieve the dynamic property from the property source instance and display the dynamic property. Nowhere in Hirsch is there even a remote reference to a control receiving both a reference to an object instance and a property source instance, and then retrieving the dynamic property from the property source instance and displaying such a property. Instead, Hirsch merely describes an object inspector

window 30 that displays an object's properties and events. Hirsch fails to teach or allude to any control whatsoever or such a control receiving references to multiple object instances (i.e., a property source instance and an object instance) as claimed.

In addition to the above, the Office Action further relies on col. 4, lines 11-34 and col. 3, lines 8-27 to teach the various limitations of claim 5. Applicants note that col. 4, lines 11-34 are not relevant whatsoever to the present claim limitations. Such text merely describes the ability for a developer to build a virtual world having a scene containing information that is linked to data stored in a database. Hirsch builds data sources using a block diagramming tool which generates database queries. Hirsch's scenes are created with a drawing editor that binds data sources to the graphical elements of the scenes. However, what is wholly and completely lacking from such text is any description of the creation of a property source instance, an object instance, a property created by the property source instance for the object instance at run time, and/or a value for the property created by the property source instance for the object instance at run time (all of which are explicit claim limitations).

In addition, to teach the association claim limitation, the Office Action relies on col. 3, lines 8-27. Such text merely supports the arguments set forth above in that Hirsch's dynamic properties are merely properties that are evaluated at run time and are not static. Such a teaching does not and cannot teach the express claim limitations relating to the creating of both a dynamic property and a value for that property at run time.

In view of the above, Applicants submit that Hirsch does not and cannot teach, disclose, or suggest the invention as claimed.

Claims 9 and 13

These independent claims are directed towards the display of a property in a property list. More specifically, once an object with a property has been retrieved, that same object provides an ActiveX control that defines a user interface for displaying and editing the property. A list of properties is created. The ActiveX control is used to display a user interface for one of those properties in the list. Thus, multiple ActiveX controls are used to display individual properties in a property list.

Applicants submit that Hirsch clearly fails to teach, disclose, or suggest such unique attributes as set forth in the present claim limitations. In rejecting these claims, the office Action asserts that Hirsch's calendar control teaches the invention as claimed (col. 12, lines 8-31). Col. 12, lines 8-31 describes a 2-column entry form for setting the values of a selected object's properties. The left column displays the name of the property while the right column is a read/write column that displays the property's value. The values entered can be constants or can be calculated containing functions or parameters or column names from a data source. If a property value is of an enumerated type, a drop down combo box may be displayed listing the legal values. Further, if the property is a date or time, a calendar control may be displayed beneath the property value when the field is active.

However, what should be noted is that nowhere in Hirsch is there any description that the object itself provides the custom control to display the properties of the object. Instead, Hirsch explicitly teaches a defined set of controls that are used independent of the object. In this regard, Hirsch's object does not provide the combo box that is used to display a list of enumerated properties. Instead, an object inspector provides a list of properties and determines what controls are used to edit the properties. Such controls are thus provided by the object inspector and not the object whose properties are being displayed in the tabbed sheet (as claimed). In this regard, the object inspector's properties are not being displayed in the tabbed sheet. Instead, the object inspector is used to display a different object's properties.

In contrast to Hirsch's teaching, the presently claimed invention provides that the object provides a custom control that defines a user interface for displaying and editing one of its own properties. Such a capability is wholly and complete missing, both explicitly and implicitly from Hirsch.

Further, Hirsch does not even remotely describe that the control is a custom control. Instead, Hirsch's controls are statically defined based on the type of property being displayed (e.g., if the property value is of an enumerated type, a drop-down combo box is used or if it is a date or time, a calendar control is used). Such a control is not a custom control provided by the object and used to edit a particular property of the object at the choice of the object itself.

Moreover, the various elements of Applicants' claimed invention together provide operational advantages over Hirsch. In addition, Applicants' invention solves problems not recognized by Hirsch.

Thus, Applicants submit that independent claims 1, 5, 9, 13, and 17-18 are allowable over Hirsch. Further, dependent claims 2-4, 6-8, 10-12, and 14-16 are submitted to be allowable over Hirsch in the same manner, because they are dependent on independent claims 1, 5, 9, 13, and 17-18, respectively, and thus contain all the limitations of the independent claims. In addition, dependent claims 2-4, 6-8, 10-12, and 14-16 recite additional novel elements not shown by Hirsch.

It is believed that no fees as due at this time. Nonetheless, should any charges be deemed necessary, please charge any such fees to, or credit any overpayments to, Deposit Account No. 50-0494 of Gates & Cooper LLP.

VI. Conclusion

In view of the above, it is submitted that this application is now in good order for allowance and such allowance is respectfully solicited. Should the Examiner believe minor matters still remain that can be resolved in a telephone interview, the Examiner is urged to call Applicants' undersigned attorney.

Respectfully submitted,

GATES & COOPER LLP
Attorneys for Applicant(s)

Howard Hughes Center
6701 Center Drive West, Suite 1050
Los Angeles, California 90045
(310) 641-8797

Date: November 7, 2007

By:/Jason S. Feldmar/
Name: Jason S. Feldmar
Reg. No.: 39,187

JSF/sjm/kmk

G&C 30566.296-US-U1